This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

A Generic Framework for Constraint-Driven Data Selection in Mobile Crowd Photographing

Huihui Chen, Bin Guo, *Senior Member*, Zhiwen Yu, *Senior Member*, Liming Chen, *Member*, and Xiaojuan Ma, *Member*

***Abstract*—Mobile Crowd Photographing (MCP) is an emerging area of interest for researchers as the built-in cameras of mobile devices are becoming one of the commonly used visual logging approaches in our daily lives. In order to meet diverse MCP application requirements and constraints of sensing targets, a multi-facet task model should be defined for a generic MCP data collection framework. Furthermore, MCP collects pictures in a distributed way in which a large number of contributors upload pictures whenever and wherever it is suitable. This inevitably leads to evolving picture streams. This paper investigates the multi-constraint-driven data selection problem in MCP picture aggregation and proposes a pyramid-tree (PTree) model which can efficiently select an optimal subset from the evolving picture streams based on varied coverage needs of MCP tasks. By utilizing the PTree model in a generic MCP data collection framework, which is called CrowdPic, we test and evaluate the effectiveness, efficiency and flexibility of the proposed framework through crowdsourcing-based and simulation-based experiments. Both the theoretical analysis and simulation results indicate that the PTree-based framework can effectively select a subset with high utility coverage and low redundancy ratio from the streaming data. The overall framework is also proved flexible and applicable to a wide range of MCP task scenarios.**

***Index Terms*—Picture stream, mobile crowd photographing, data selection, pyramid tree, constraints.**

I. Introduction

LARGE-SCALE sensing is the key to the success of ubiquitous human-machine systems. The increasing prevalence of smart devices and their inherent mobility led to the rapid emergence and adoption of a novel large-scale sensing paradigm, namely Mobile Crowd Sensing and Computing (MCSC) [1]. MCSC utilizes the power of users to accomplish specific sensing tasks without requiring pre-deployed dedicated infrastructure. It is thus a typical human-machine system with the participation of human in large-scale data collection. It can collect information of interest in remote physical environments by recruiting smart device users.

MCSC can make use of different modalities of sensing, e.g. picture taking, audio recording and GPS logging. Among these modalities, Mobile Crowd Photographing (MCP) that uses built-in cameras of smart devices has become a predominant sensing paradigm. Previous research and applications, e.g. CreekWatch [2], GarbageWatch [3], SmartEye [4], PhotoCity [5], WreckWatch [6], FlierMeet [7], Mediascope [8], iMoon [9] and SakuraSensor [10], indicated that MCP is useful and superior to traditional approaches in visual sensing, e.g. deployment of static cameras for monitoring.

Data collection of an MCP application is usually conceptualized as a *task* in a traditional multi-task crowdsourcing platform, such as Amazon's Mechanical Turk and Medusa [12]. In this way, an MCP application can be characterized as a process involving several roles. This process typically includes 1) a *data requester* initiates and publishes *a task* by specifying the requirements to the centralized task management server; 2) *workers* participate the task according to task requirements and their characteristics; 3) *worker*s acquire and submit *data* to the platform; and 4) the collected data are processed (e.g., for trust and quality of data) before returning to the *data requester*.

The requirement specification in an MCP application is key to defining a sensing task, and also, the most challenging effort since each MCP task is different in terms of their sensing targets (e.g. buildings [5], flyers [7] and event [28]) and coverage requirements (e.g. the target sensing area and time period, single-shot or multi-shots including different shooting directions [18]). For example, pictures taken from different shooting angles can be useful for complex applications, e.g. 3D modeling [5] or event sensing [28], while single-shot sensing is often adopted for monitoring, e.g., the status of garbage [3].

Existing MCP systems usually only support one specific task (e.g. river pollution monitoring [2], climate change sensing [11], and disaster/event picture collection [4, 8, 13]). This leads to the reusability and scalability limitation problems as these systems are application-dependent. To solve these problems, we have developed a generic picture collection framework, which has the following improvements but also challenges.

Firstly, for the data requester, this framework facilitates the rapid specification of MCP tasks taking into consideration of different constraints, eliminating the need to develop domain-specific, application-dependent proprietary systems.

This work was partially supported by the National Basic Research Program of China 973 (No. 2015CB352400), the National Natural Science Foundation of China (No. 61602230, 61332005, 61373119).

H. Chen, B. Guo and Z. Yu are with Northwestern Polytechnical University, Shaanxi, China, 710129 (e-mail: {chenhuihui.cn, guobin.keio, zhiweny}@gmail.com).

L. Chen is with De Montfort University, UK (e-mail: liming.chen@dmu.ac.uk).

X. Ma is with the Hong Kong University of Science and Technology, China (e-mail: mxj@cse.ust.hk).

Copyright (c) 2012 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

This will lower the barrier for ordinary users to post MCP tasks and meet their personalized needs, e.g., a botanist wants crowdsourcing pictures of some plants [11]. However, different MCP tasks have distinct sensing targets and constraints (e.g. where and when to sense, sampling frequency, single or multiple shooting angles). In order to build this generic MCP framework, we make a thorough analysis of MCP concepts and needs of applications to comprehensively model different tasks.

Secondly, for the worker, this framework provides them with a unique entrance to participate in different MCP tasks, which can simplify worker recruitment and facilitate task query or recommendation. Moreover, one worker can take multiple tasks at the same time to earn more rewards. However, a worker may upload pictures that are similar to others' because workers execute tasks in a distributed and non-cooperative manner, so pictures carried by different workers might be redundant. Therefore, picture selection is thus important to MCP tasks. First of all, uploading pictures is traffic-consuming. Through uploading a thumbnail of the picture for redundancy detection first, pre-selection approach (i.e., selecting pictures before they are uploaded) can be leveraged to save traffic [24]. Secondly, pictures contributed by workers for a task will construct a picture stream. In order to make a real-time decision on whether a full-size picture should be uploaded, the framework should support online clustering of the data stream to recognize redundant pictures. The redundancy is characterized by the predefined constraints and the collected data (i.e., the so-called *situation*) of the task. Therefore, we should develop a constraint-driven data stream clustering approach for the optimized picture selection and efficient picture collection.

In order to deal with the above issues, we developed a generic participatory picture data collection framework called CrowdPic. CrowdPic is applicable to tasks of varying themes and constraints, allowing the data requester to specify the requirements and constraints on picture collection from multiple dimensions, such as time, locations, directions, multi/single-shot, and frequency. In addition, it leverages a data selection method that can analyze and select an optimized subset of user-contributed data online from the original picture stream. In particular, we made four contributions as follows:

1) *Providing a formal, generic, conceptual MCP framework by analyzing the procedure of picture collection.* Based on data collection requirements of existing single-task MCP applications and the sensing capability of smart mobile devices, we are the first to utilize a generic MCP framework and to analyze its related issues, such as how to define an MCP task, how to utilize embedded sensors, how to measure the redundancy of a picture, and so on.

2) *Giving a formal formulation of the optimal data selection problem for MCP*. Since distributed participants might contribute redundant pictures, we define the selection problem as choosing the most diversified pictures to obtain the maximum coverage based on the predefined constraints. It can be viewed as an extension of the vertex independent set covering problem [26], and it is also considered as the optimal solution of picture selection to evaluate our method's acting on the picture stream.

3) *Developing a pyramid-tree (PTree) model to enable efficient near-optimized data selection through clustering the picture stream*. The PTree model and its associated tree generation rules allow the framework to intelligently cluster the streamed pictures according to the task constraints and the situation of available data. The clustering result facilitates the decision-making process on picture acceptance/rejection.

4) *Developing metrics and measurements to validate the performance of the picture selection method*. For the purpose of evaluating the recall and the precision, we design new measurements to evaluate the true positive picture subset when there are many equivalent picture subsets. We further conduct theoretical analysis and crowdsourced dataset-based experiments to evaluate the performance of the framework. Results show that PTree-based selection method achieves a better trade-off between efficiency and effectiveness.

The rest of the paper is organized as follows. Section II outlines related works on MCP systems and picture collection/selection methods. Section III presents formal MCP process and role modeling followed by the description of the generic MCP framework in Section IV. Section V describes the PTree-based clustering and data selection methods. We present and discuss experiment results in Section VI, and conclude the paper along with the speculation of the future work in Section VII.

II. Related Work

A. MCSC and MCP

MCSC has become a hot research topic in the field of ubiquitous computing. Ma et al. [17] investigated the opportunistic characteristics of human mobility from the perspectives of both sensing and data transmission and presented approaches to collect MCSC data more efficiently. Zhang et al. [16] exploited the 4W1H – a four-stage life cycle, to characterize the MCSC process. Key techniques on MCSC include methods on user privacy and data trustworthiness issues [15], proper worker selection [19], and incentive mechanisms [20].

MCP has become a dominating method of MCSC, which collects information and extracts knowledge from crowd-contributed pictures. Typical MCP applications include monitoring the pollution of creeks [2], detecting traffic signals in urban areas [21], reposting and sharing fliers distributed in residential communities [7], extracting prices of goods in the market [22], gathering pictures of buildings for 3D city modeling [5], sensing event in real time [28] and reporting the scenes in emergency situations [4, 13, 14].

Most of these MCP applications mentioned above is single-task and only collect one kind of pictures. It usually uses specific picture selection criteria according to its goal to abandon redundant and low-quality pictures (e.g., too dark, or motion-blurry) [4, 7, 13, 14, 15]. Different from existing MCP applications, CrowdPic is a multi-task framework that can collect different kinds of pictures for various tasks and automatically filters them at the same time according to diverse constraints defined in tasks by picture users.

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

B. Picture Selection in MCP

It is important to select a subset of pictures which have the same coverage (according to task constraints) with the whole data stream while reducing demands on resources. There are different criteria for picture selection in MCP applications. Table I summarizes these representative MCP applications and criteria used in their data selection. In our work, we view data collections of these MCP applications as tasks with different sensing targets and constraints and build a generic data collection framework to meet diverse MCP application constraints/requirements. To the best of our knowledge, this is the first work towards this direction.

TABLE I
SELECTION CRITERIA AND RELATED APPLICATIONS.

Criteria	Applications
Multiple directions	SmartPhoto[23], PhotoCity[5], InstantSense[28].
Single direction	FlierMeet[7], MobiShop[22].
Local	GarbageWatch[3], WreckWatch[6], InstantSense.
Global	CreekWatch[2].
Small time slot	SmartEye[4], MediaScope[8], InstantSense.
Large time slot	Climate [11], PhotoCity[5], iMoon [9].
Refreshing slowly	FlierMeet[7], Climate [11], SakuraSensor [10].
Refreshing quickly	SignalGuru[21], GarbageWatch[3], WreckWatch[6], Jameye[12].

The diversity of selected pictures is deemed as a key indicator of sensing quality by an MCP application. In order to select diverse pictures for obtaining wider perspectives, MCP applications usually use one of the following two criteria to measure pictures' similarity. One is to assess pictures' *visual contents*, e.g., FlierMeet [7], GarbageWatch [3], SmartEye [4] and MobiShop [22]. The other is to assess pictures' *semantic contents* based on photographing contexts of pictures, e.g., InstantSense [28] selects pictures of different sub-events in order to generate the visual summary of an event through high-relevant and low-redundant pictures. In this paper, CrowdPic will assess pictures' similarity based on the combinations of pictures' visual and semantic features.

There are different methods to calculate pictures' similarity distance according to one or two features of the picture for redundancy detection. For example, FlierMeet [7] uses SIFT-based visual distance for pictures' similarity measurement. SmartPhoto [23] exploits the shooting angle to characterize redundancy. InstantSense [28] assesses pictures' similarity according to human's photographing behavior. These applications different methods to compute similarity. In order to adapt to different MCP application requirements, the generic data collection framework, e.g. CrowdPic, should be able to conquer the limitation of fixed distance methods and calculate the partial similarity of the pictures according to different features. To this end, CrowdPic treats both distance calculation methods and thresholds as variables, and different methods can be used in the similarity measurement process.

Picture selection can be carried out in different phases based on various considerations such as technical limitations. For instance, some applications, e.g. GarbageWatch [3], manually filters redundant pictures offline until the whole picture sets were obtained (i.e., data collection tasks were over), while some applications, e.g. FlierMeet [7], automatically filters redundant pictures online during the picture collection process in a pre-selection way. As discussed in the introduction, to save traffic cost and select representative data to be uploaded based on task constraints, CrowdPic adopts the online picture selection approach by developing a novel PTree method to cluster picture streams with constraints.

III. SYSTEM MODELING AND PROBLEM FORMULATION

A. Four Stages of MCP tasks

By analyzing the existing MCP applications, an MCP task can be characterized by a generic four-stage process, as depicted in Figure 1: *task initiation*, *task execution*, *data aggregation* and *result handover*. At the *task initiation* stage, *data requester*s define their tasks with different requirements and the *task management server* assigns them to suitable workers. At the *task execution* stage, *worker*s take pictures according to task requirements and upload them to the *backend server*. As the server receives pictures uploaded by distributed workers intermittently, it inevitably contains redundant information. As such, at the *data aggregation* stage, it is necessary to group and select pictures from the picture stream based on task specifications. In the *result handover* stage, the data repository is made available to the data requester upon task completion.

As duplicate pictures can lead to unnecessary data traffic in MCP applications, an approach to solving this problem as described in [24] is that the thumbnail and related contextual

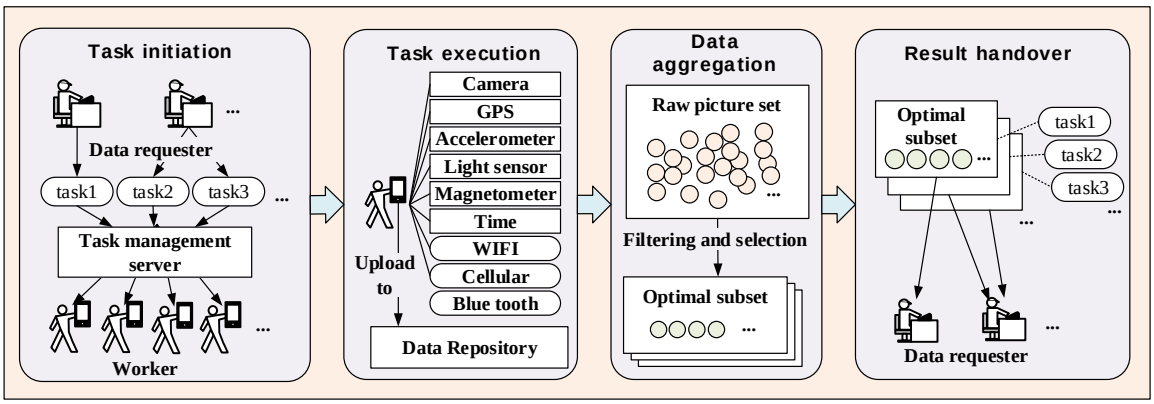


Fig. 1. Four Stages of MCP tasks.

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

information of a picture are first uploaded and analyzed at the server side based on the MCP task requirements. The analysis result can be used to determine whether the full-sized picture is needed, which is the *pre-selection* for MCP. We followed this approach in this study for *picture aggregation*. It is expected that such a decision will be made immediately once a thumbnail is uploaded.

Based on the above discussion, we can identify three requirements for building a general framework for participatory MCP data collection, as presented below.

1) *A multi-facet task model for varied MCP task specifications, which can be adapted to different picture collection requests and constraints*.

2) *An efficient picture selection approach to deciding whether a full-sized client-carried picture should be submitted or deleted*.

3) *Maintaining the coverage of selected data, i.e. quality of sensing, when selecting data from the streamed pictures according to the task specifications*.

B. MCP Task Modeling

To adapt to various MCP task publication, a generic MCP framework requires a flexible and multi-facet task model which can define tasks with different types of demands and constraints. We conceptualize the CrowdPic task model and its associated elements as below.

1) The Task Model

A CrowdPic task can be modeled in two elements, namely *task specification and task description*. The former element allows data requesters to define multi-dimensional constraints for picture collection and selection, and the latter allows workers to easily understand and execute a task and is usually described in natural language.

TABLE II
DEFINITION OF TSK

Symbol	Definition
tid	The identifier of the MCP task.
whn	A time span defined by the start time *TS* and the end time *TE* of a task.
whr	A geographical area defined by GPS points specified on the digital map (e.g. Google map) for performing the task.
vlmn	The desired volume of the picture set.
cycl	A time span denoting the changing or refreshing cycle of the sensing target.
grid	A geographical distance within which the same target or similar targets might be seen.
mView	A numerical value in [0,π), which denotes the multi-view photographing constraint with the angle of two 3D shooting directions.
imgSim	The method used to detect similar images, e.g. SIFT [27].

In order to collect highly relevant data, the task specification module uses quantifiable parameters to guide picture collection. Based on the analysis of existing MCP applications, we conceive a 7-element data structure to denote an MCP task (TSK): {*whn, whr, vlmn, cycl, grid, mView, imgSim*} for MCP task specification which is described in Table II. Overall the 7-tuple specifies multiple constraints in a picture collection task. The first three items *whn*, *whr*, and *vlmn* characterize the generic information about a task. The remaining items are constraints for picture selection at the backend server.

In the following, we use an example to illustrate the usage of the task model. Suppose that we want to gather the poster information for Christmas sales in a shopping area, we can use an MCP application like FlierMeet [7]. The task can be formalized as {*whn*=(20151210 to 20151225), *whr*=(34.64, 112.42) to (34.65, 121.41), *vlmn*=2000, *cycl*=5(day), *grid*=20(meter), *mView*=π/2, *imgSim*=(SIFT[27], high)}. This task can be interpreted as: this task needs to recruit workers to take 2,000 pictures within the geographical area between (31.29, 121.47) to (31.09, 120.97) from Dec. 10 to Dec. 25, 2015. The task constraints for determining redundant data is that (i) pictures are taken within 20 meters and 5 days, (ii) the angle of their shooting directions is less than π/2, and (iii) the visual similarity measured by SIFT features is at a high level.

2) MCP Picture Model

When the worker takes a picture, the mobile client will save the image file and its associated context information. The MCP picture is modeled using a 10-element data structure PIC: {*pid*, *tid*, *wid*, *img*, *tp*, *ta*, *sDir*, *loc*, *light*, *acc*}. Each item is explained in Table III below where *pid*, *tid*, and *wid* are identifiers and are not used as features of a PIC in the following. The associated context information of photographing can also be utilized to match task constraints for the future picture selection. For example, *light* and *acc* are used for context-based image quality evaluation, which has been discussed in [7].

TABLE III
DEFINITIONS OF PIC'S ITEMS.

Sym.	Feature	Definition
pid	-	The identifier of the PIC.
tid	-	The identifier of the MCP task that the PIC is taken for.
wid	-	The identifier of the worker.
img	Image	A full-size image and its thumbnail saved by the mobile client App using different image resolution configurations. The small-size thumbnail *img_t* is uploaded first, and if the picture is selected, then the full-size one will be uploaded.
tp	Timestamp	The time point of a photographing.
ta	Time of being uploaded	The time point that the picture arrives at the data center.
sDir	Direction	The shooting-direction of a picture defined by the three angle vectors <*azimuth, pitch, roll*> which can be calculated on the basis of observations of the accelerometer and the magnetometer [25].
loc	Location	A GPS coordinate that denotes the location of photographing.
light	Light	The ambient light level observed by the light sensor.
acc	Motion	3D accelerometer reading values at the moment of photographing to assess the motion-blurry image.

3) Relationship between TSK and PIC

After a task is defined, its *task description* will be pushed to worker candidates. Then those people who accept this task will be considered as workers of this task and TSK will be downloaded. In order to collect high-quality pictures, uploaded pictures must meet tasks' requirements. Thus, first of all, pictures will be roughly checked by mobile devices for whether those pictures are taken within the right time range (*whn*) and in the right area (*whr*) based on the downloaded TSK. Secondly, partial PIC will be uploaded to the data management server based on the enabled items of the task specification. For

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

example, if the TSK only values *mView*, *grid*, *imgSim*, then those corresponding items of PIC, i.e. *tid*, *sDir* (for *mView*), *loc* (for the *grid*), *img_t* (for *imgSim*), will be uploaded to the data management server. At last, if a picture is validated as good quality by the server, then the full-size image will be further uploaded [24]. Here, uploaded items of PIC can be customized to make our framework more adaptable and energy-saving.

4) Picture Stream

The picture stream $\mathbb{X}$ consists of a series of pictures $\mathbb{X}=\{X_1, \ldots, X_k, \ldots\}$ for a certain task arriving at the data center at time points $ta_1, \ldots, ta_k, \ldots$, and each $X_i \in \mathbb{X}$ is a multi-dimensional record denoted by $X_i=<x_{i,1},\ldots,x_{i,d}>$, where each x is an item from PIC in Table II, and d varies according to the task definition. For example, the task requester only needs PICs that are composed of images and their locations, then $X_i \in \mathbb{X}$ will only save *img* and *loc* (besides *pid*, *tid* and *wid*) of a PIC and d=2.

Since the picture has heterogeneous features, we use a Boolean function to measure the duplicate relationship of two pictures X_i, X_j, which is calculated by the function $\mathcal{D}$ in Eq. (1).

$$\mathcal{D}(X_i, X_j) = \wedge_{k=1\ldots d} \mathcal{H}\left(x_{i,k}, x_{j,k}\right), \quad (1)$$

where $x_{i,k} \in X_i$, $x_{j,k} \in X_j$, and Boolean function $\mathcal{H}$ calculates whether two sub-items are similar. The calculation method of $\mathcal{H}$ can vary for different k. For instance, if $x_{*,k}$ denotes locations, then $\mathcal{H}$ is a method to determine whether two locations are close enough to take similar pictures or not.

C. Maximum Coverage Problem

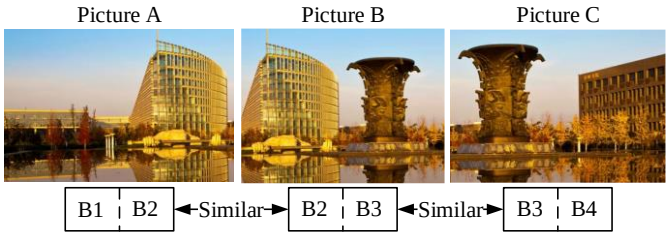
The purpose of picture selection is to find a minimal subset from streamed pictures that meets an application's multiple-coverage requirements with little data redundancy. This can be formalized as finding a subset with the maximum coverage in terms of subset utility. In this case, the max-coverage optimization problem of picture selection can be formulated as finding Maximum Independent vertex Set (MIS) of a graph.

Definition 1 Maximum Independent Set (MIS): A subset S of the vertex set V of a graph G is called independent if no vertexes of S are adjacent in G. $S \subseteq V$ is a maximum independent set of G if G has no independent set S' with $|S'|>|S|$ [26].

Firstly we model the picture sets as an undirected and unweighted graph G:<$\mathbb{X}$, $\mathbb{E}$>, where pictures from the picture stream $\mathbb{X}$ are vertexes and pictures' similarity relationship forms the edge set $\mathbb{E}=\{(X_i, X_j)$: if $\mathcal{D}(X_i, X_j)$=True, $X_i \in \mathbb{X}$, $X_j \in \mathbb{X}\}$. In order to select diverse pictures to maintain the quality of sensing, we define the Maximum Diversified Subset (MDS) $\mathbb{M} \subseteq \mathbb{X}$ of the picture stream $\mathbb{X}$ as follows.

$$\mathbb{M} = argmax\{|\mathbb{M}|: \mathcal{D}\left(X_i, X_j\right) = \text{False}, X_i, X_j \in \mathbb{M}\}. \quad (2)$$

As shown in Figure 2, four buildings are recorded by three pictures, the picture set {A, C} has the maximal coverage to the whole set and it is the MDS, and it is also the MIS of G. Therefore, finding the MDS of the picture set $\mathbb{X}$ is equal to find the MIS of the graph G: <$\mathbb{X}$, $\mathbb{E}$>. If the picture stream is A-B-C, then B will be discarded, but if the picture stream is B-A-C, then only B will be collected. It is difficult to obtain the optimal selection from the picture stream and it is also NP hard for a complete dataset. In the following, we will propose the framework to obtain the near-optimal MDS.



Picture {A,C} has the maximal coverage to the scence.

Fig. 2. Pictures, the graph, and MDS.

IV. The CrowdPic Framework

Based on the identified requirements and problem analysis, we develop the CrowdPic framework, which is described below.

A. An Overview

The CrowdPic framework, as shown in Figure 3, mainly addresses the optimal picture selection in two MCP stages shown in Figure 1: *task execution* and *data aggregation*.

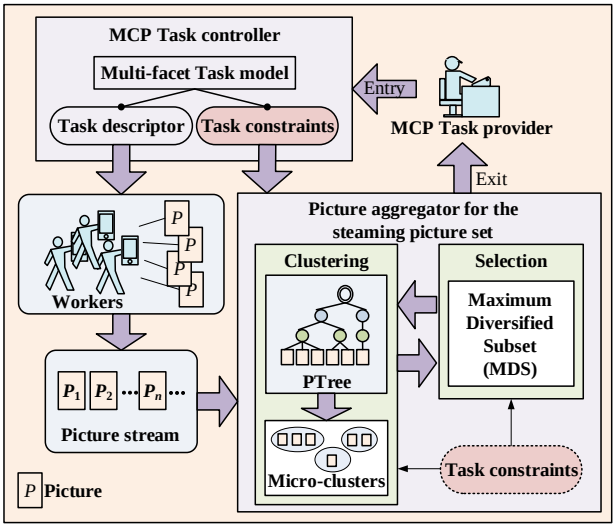


Fig. 3. The CrowdPic framework.

As mentioned in Subsection III-*B*-1, a sensing task contains a readable task descriptor and a set of task constraints. The *task controller* is responsible for assigning a task to a group of qualified workers according to the task needs. Picture-taking tasks usually require workers to be at the scene, but it is impossible to always recruit workers at the exact scene defined by the task. Therefore, we have to ask workers to go to defined places to take pictures, and this process is called task assignment and worker incentive, which is out of the scope of this work. The *picture aggregator* module collects and selects pictures from the picture stream in view of predefined task constraints. We have proposed a Pyramid Tree (PTree)-based data stream clustering method in CrowdPic to dynamically group redundant pictures. After being clustered, the data stream is divided into many micro-clusters and then the MDS is

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

composed of elements from each of these microclusters. Note that the given task constraints always hold during the MDS construction process and the MDS can get updated when a new picture arrives. So the picture aggregator module is crucial to CrowdPic and we will present its detailed workflow below.

B. The Workflow of the Picture Aggregator

PTree-based clustering is the core component of the *picture aggregator* in Figure 3. As shown in Figure 4, PTree-based clustering contains two major parameters: *layering mapping* (LM) and *branching parameters* (BP).

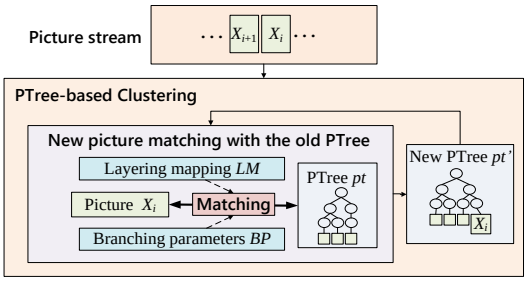


Fig. 4. A new arrival picture X_i matches with the PTree *pt*. The PTree produces a new branch when a new leaf node is created according to the matching result.

LM denotes the one-to-one mapping between features of a picture (i.e. items of PIC) and layers of the tree. For example, an MCP task needs pictures with features $\mathbb{F}$={location, timestamp, image}, then the LM *lm*={location, image, timestamp} means that *loc*, *img*, and *tp* of PIC are used as parameters $x_{i,1}$, $x_{i,2}$, $x_{i,3}$ respectively for the function $\mathcal{H}$ in Eq. (1). In other word, layers from top to bottom of the PTree are related to the location, image, and timestamp of this task's pictures. Note that features' order in $\mathbb{F}$ is different from those in *lm*, so there are *d*! LMs in total if $|\mathbb{F}|=d$. Different LMs can affect PTree performance, we will discuss this later.

A BP $\mathbb{B}=\{bp_i: i=1,..,|\mathbb{F}|\}$ contains a set of parameters to calculate function $\mathcal{H}$ in Eq. (1). Each $bp\in\mathbb{B}$ is a 2-tuple <*d_mthd*, *d_th*> including the method of similarity measurement denoted by *d_mthd* and the corresponding threshold denoted by *d_th*. Both *d_mthd* and *d_th* are related to the corresponding feature. For instance, bp_1=<Euclidean distance, 50m> when $\mathbb{F}$={location,…}, which means if the Euclidean distance of calculated terms is less than 50m, then the function $\mathcal{H}$ returns true.

A pyramid tree (PTree) is initially empty and when pictures arrive one by one, the PTree will grow a new leaf for each coming picture. The position of a new leaf in the PTree is determined by how the corresponding PIC matches the existing tree based on LM and BP. Which picture will be selected depends on where these pictures are in the PTree. Therefore, finding the position for each picture is the most important process of the PTree-based clustering. In the next part, we will describe the general concepts and features of the PTree and use examples to explain the working mechanism of PTree-based clustering and selection.

C. Definition and Attributes of a PTree

1) Definition of a PTree

PTree is a (*d*+2)-layer hierarchical tree structure as shown in Figure 5. Its leaf nodes are in the bottom layer and it is generated according to a data stream $\mathbb{X}$ whose element $X_i\in\mathbb{X}$ has *d* dimensions. Its root node is in the 0-th layer and the other layer is denoted as *p*-th layer ($1\le p\le d+1$). For a given picture stream $\mathbb{X}=\{X_1,\ldots, X_{12}\}$, if each picture has 3 dimensions, i.e. $X_i=<x_{i,1}, x_{i,2}, x_{i,3}>$, then a 5-layer PTree will be generated – as illustrated in Figure 5. Based on this example, we will introduce the attributes of a PTree.

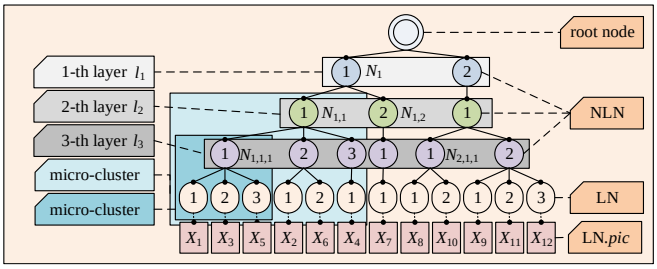


Fig. 5. A 5-layer pyramid tree. The number in the circle represents the number of a tree node. NLN is abbreviation of non-leaf node, LN is abbreviation of leaf node, and LN.*pic* is an attribute and refers to a picture.

2) Attributes of the PTree

All nodes of the PTree have at least two attributes: the number denoted by *no* and the identifier denoted by *id*. The number *no* is the serial number of a node in its siblings ($no\ge 1$), represented as numbers in the circles/nodes in Figure 5. The identifier *id* is formed by the sequence of *no*s of the nodes in its path, indicating the path from the root node to the node itself, e.g. *id*='1,1,1' for the node $N_{1,1,1}$ in Figure 5.

For a leaf node, it has an attribute *pic* to refer to the picture of the node. Pictures in sibling leaf nodes are similar. Based on the first-uploaded-first-selected rule, only pictures in the Leaf Nodes (LNs) whose *no*s are 1 will compose the selected subset. For instance, pictures X_1, X_2, X_4, X_7, X_8 and X_9 in Figure 5 are selected as valuable ones.

The picture stream can be divided into a lot of micro-clusters through dividing leaf nodes of a PTree into a number of micro-clusters which are composed of pictures in its offspring LNs.

Non-Leaf Nodes (NLN) in different layers have different scale micro-clusters. As shown in Figure 5, the subset $\{X_1, X_3, X_5\}$ is the micro-cluster of the node $N_{1,1,1}$ and the subset $\{X_1, \ldots, X_6\}$ is the micro-cluster of the node $N_{1,1}$.

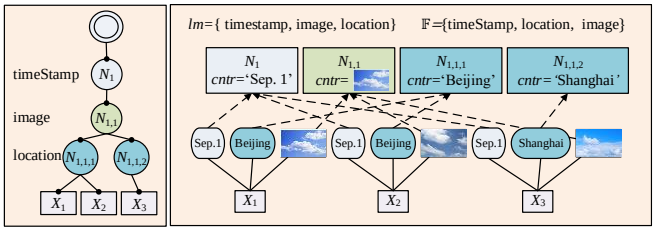
Micro-clusters of a PTree reflect the divide and conquer technique, and which feature is used to divide the picture stream first can be configured in the LM. For example, the picture stream can be first roughly partitioned based on their spatial feature (i.e. the first element in the LM is *location*) or based on their temporal feature (i.e. the first element in the LM is a *timestamp*) as well.

In order to generate a PTree, each NLN has the attribute *cntr*, which denotes the centroid computed based on this NLN's micro-cluster. For the NLN on the *i*-th layer, assuming that the *i*-th layer corresponds to the *k*-th feature of the picture, then the NLN's Feature-Related Micro-Cluster (FRMC) consists all *k*-th sub-items of pictures in its micro-cluster. The attribute *cntr* refers to the center of the FRMC of an NLN. Two methods are used to calculate *cntr*: (i) The method First-as-Center (FaC) selects the first item (i.e. oldest item) of a micro-cluster; (ii)

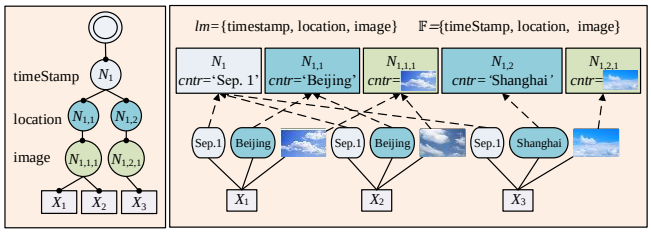
2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

Last-as-Center (LaC) selects the last item (i.e. latest item) of a micro-cluster. Next, we use an example to illustrate the PTree and LM's effect.



(a) Using the LM *lm*={timestamp, image, location} and FaC.



(b) Using the LM *lm*={timestamp, location, image} and LaC.

Fig. 6. Two PTrees generated from pictures of weather monitoring.

As shown in Figure 6, two PTrees are generated according to the same picture stream $\{X_1, X_2, X_3\}$, the same BP and two different LMs. The PTree in Figure 6 (a) has four NLNs, which is less than that of the PTree in Figure 6 (b), so the LM can impact the PTree generation. In order to effectively utilize the PTree model in the picture selection process, we introduce the PTree generation rules and analyze the storage and computing cost in following sections.

In conclusion, a leaf node (LN) is defined as a 3-tuple structure <*no*, *id*, *pic*>, and a non-leaf node (NLN) is defined as another 3-tuple structure <*no*, *id*, *cntr*>.

D. Creating a PTree

Finding the position for each picture is the main step of PTree generation and the most important step of picture selection. Next, we will introduce parameters and rules for PTree generation.

1) Matching Algorithm of PTree Generation

Function $\mathcal{H}$' in Eq. (3) is used to assess whether the picture X_i will be in an offspring LN of an NLN *nln* in the *p*-th layer. If $\mathcal{H}$' returns true, then *nln* is a Matched NLN (MatNLN) of X_i in the *p*-th layer.

$$\begin{cases} \mathcal{H}'(X_i, nln) = True: & \text{if } \mathcal{Q}(x_{i,r}, nln.cntr) \le d\_th_r \\ \mathcal{H}'(X_i, nln) = False: & \text{Otherwise} \end{cases}, \quad (3)$$

where the *r*-th feature of X_i is set to be related to the *p*-th layer in the LM, and function $\mathcal{Q}$ calculates the distance by using the distance measurement method d_mthd_r and threshold d_th_r of the *r*-th feature, i.e., bp_r=<d_mthd_r, d_th_r>.

2) Branching Algorithm of PTree generation

The PTree generation relies on continuously branching guided by a set of rules. When a *d*-dimension picture X_i is uploaded, it finds its MatNLNs and will be in the new LN of the newly created branch, as shown in Algorithm 1. At the beginning, *Nc* is the root node and *p*=1.

Algorithm 1: **FindMatNLN** (*Xi*, *Nc*, *p*, *d*)

```
// Xi denotes the picture, p denotes the number of the current
   layer, Nc denotes the current tree node.
If p = d Then
  Create a leaf node of Nc, and Xi is stored in this leaf node.
Else
  Find ← False;
  For each node N in Nc.childNodes Do
    If H'(Xi, N) = True then
      Find ← True;
      FindMatNLN (Xi, N, p+1, d);
      Break;
    End if
  End for
  If Find = False Then
    While p <= d-1 Do
      Create a child node N of node Nc on the (p+1)-th layer;
      Nc ← N; p++;
    End while
    Create a leaf node of Nc, and Xi is stored in this leaf node.
  End if
End if
```

V. The Performance of PTree- Based Selection

This section introduces factors which impact the performance of the PTree generation and metrics for measuring the coverage of the selected pictures.

A. Efficiency Affected by PTree's Shapes

Given different LMs and BPs, the shape of the PTrees generated from the same picture stream might be different under different branching/layering processes. As shown in Figure 7, there are three basic shapes of a PTree as follows:

- An I-shape PTree has only one node (the root node or an NLN) that has a large number of child NLNs.
- An inverted-T shape (iT-shape) has only one node (the root node or an NLN) that has all LNs.
- Most NLNs in an A-shape PTree have more than one child node and their numbers of child nodes are slightly different.

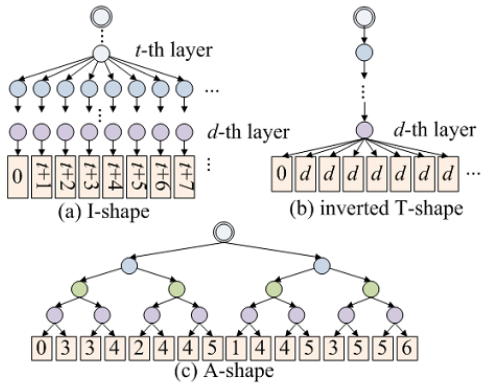


Fig. 7. Three basic PTree shapes. The Number in the leaf node denotes *AccNLN* of the picture linked with this leaf node. For presenting *AccNLN*, a binary tree is used to represent an A-shape PTree here.

During the PTree generation process, the computation cost of identifying each picture's corresponding micro-cluster is mainly arising from searching for MatNLNs. Assessing whether one NLN is a MatNLN is equivalent to calculating $\mathcal{H}$' once in Eq. (3). Therefore, we use the number of NLNs used to search MatNLNs as the estimated computation cost of

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

generating the PTree. $ComC_n$ denotes the computation cost of generating an n-LN PTree and $AccNLN_n$ refers to the number of NLNs assessed by the n-th picture X_n when searching its MatNLNs, then $ComC_n$ can be estimated by Eq. (4).

$$ComC_n = ComC_{n-1} + AccNLN_n, \quad (4)$$

where $AccNLN_1$=0 and $ComC_0$=0. *ComC* refers to $ComC_{|\mathbb{X}|}$ in the following.

B. Equivalency of Multiple Solutions

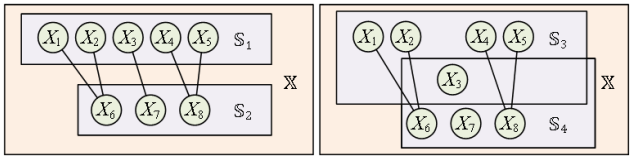
Sometimes, we might get more than one MDS, and any of them can be the optimal one, e.g., any MIS (i.e. MDS) shown in Figure 2(b) can be the optimal one. We further analyze the similarity of the selected subsets to evaluate which one is better. Given two different selection results $\mathbb{S}_i$ and $\mathbb{S}_j$, the function $\mathcal{T}$ in Eq. (5) computes their *element-similarity degree*, and $\mathcal{U}$ in Eq. (6) computes their *utility-similarity degree*.

$$\mathcal{T}(\mathbb{S}_i, \mathbb{S}_j) = \frac{|\mathbb{S}_i \cap \mathbb{S}_j|}{|\mathbb{S}_i \cup \mathbb{S}_j|}, \quad (5)$$

$$\mathcal{U}(\mathbb{S}_i, \mathbb{S}_j) = \frac{\sum_{X_m \in \mathbb{S}_i} \mathcal{L}(X_m, \mathbb{S}_j) + \sum_{X_m \in \mathbb{S}_j} \mathcal{L}(X_m, \mathbb{S}_i)}{|\mathbb{S}_i| + |\mathbb{S}_j|},$$

$$\text{where } \begin{cases} \mathcal{L}(X_i, \mathbb{S}_j) = 1, \ \exists X_k \in \mathbb{S}_j (\mathcal{D}(X_i, X_k) = True) \\ \mathcal{L}(X_i, \mathbb{S}_j) = 0, \ \forall X_k \in \mathbb{S}_j (\mathcal{D}(X_i, X_k) = False) \end{cases}. \quad (6)$$

As shown in Figure 8, $\mathbb{S}_1$, $\mathbb{S}_2$, $\mathbb{S}_3$ and $\mathbb{S}_4$ are selected subsets of $\mathbb{X}=\{X_1,\ldots,X_8\}$ and they are all independent sets. Edges of graph (a) and graph (b) are slightly different. By definition, $\mathcal{T}(\mathbb{S}_1,\mathbb{S}_2)$=0, $\mathcal{U}(\mathbb{S}_1,\mathbb{S}_2)$=1, $\mathcal{T}(\mathbb{S}_3,\mathbb{S}_4)$=1/8, $\mathcal{U}(\mathbb{S}_3,\mathbb{S}_4)$= 8/9. Though two selected subsets look like different, their utility might be close.



(a) (b)

Fig. 8. Two entire picture sets and their two selected subsets.

In order to further evaluate which selected subset is the best, the function $\mathcal{C}$ calculates the coverage of a subset $\mathbb{S}$ to the complete set $\mathbb{X}$ in Eq. (7) based on their MDS.

$$\mathcal{C}(\mathbb{S}) = \frac{|\mathcal{M}(\mathbb{S})|}{|\mathcal{M}(\mathbb{X})|}. \quad (7)$$

By definition, the MDS of $\mathbb{X}$ denoted by $\mathcal{M}(\mathbb{X})$ in Figure 8(a) is $\mathcal{M}(\mathbb{X})=\{X_1,X_2,X_3,X_4,X_5\}$, then $\mathcal{C}(\mathbb{S}_1)$=1 and $\mathcal{C}(\mathbb{S}_2)$=3/5. The MDS of $\mathbb{X}$ in Figure 8(b) is $\mathcal{M}(\mathbb{X})=\{X_1,X_2,\ldots,X_7\}$, then $\mathcal{C}(\mathbb{S}_3)$=5/6 and $\mathcal{C}(\mathbb{S}_4)$=4/6. The result of $\mathcal{C}$ is an indicator to measure the proximity of the selected subset to the optimal subset. Therefore, both subset $\mathbb{S}_1$ and $\mathbb{S}_3$ are optimal selections.

VI. Experiment and Evaluation

A. Metrics

To evaluate the algorithm, we use five basic metrics, i.e. *Covr*, *ComC*, *Redn*, *SelRt*, and *AComC*. *Covr* reflects the sensing coverage of the selected subset, which can be computed by Eq. (7). *Redn* denotes the redundancy ratio of the selected dataset, as formulated in Eq. (8), and *SelRt* refers to the ratio of the number of selected pictures to the number of raw pictures. *Covr*, *Redn*, and *SelRt* are used to evaluate the effectiveness of the algorithm. *SelRt* reflects the degree of saving the traffic and the storage. *Redn* is a negative indicator and reflects the problem of the selection result. It is the larger the better for *Covr*, but the lower the better for *Redn* and *SelRt*. *ComC* has been presented earlier in Eq. (4). And *AComC* depicts the average computation cost for each picture. Both *ComC* and *AComC* are utilized to measure the efficiency, and they are the lower the better.

$$Redn = \frac{|\mathbb{S}| - |\mathcal{M}(\mathbb{S})|}{|\mathbb{S}|}. \quad (8)$$

B. Experiment Settings

1) Distance Calculation Method

In the process of generating a PTree, methods *d_mthd*s contained in the branching parameter BP for calculating distances vary in different layers. The methods' symbols used in our experiments and their corresponding features are (Dis^{SIFT}, *Visual*), (Dis^{geo}, *Location*), (Dis^{dir}, *Direction*), and (Dis^{time}, *Timestamp*). Dis^{geo} denotes a geography distance calculation method for two *loc*s, and Euclidean distance is used in this paper. Dis^{SIFT} denotes the SIFT-based (Scale-Invariant Feature Transform based) [27] near-duplicate image matching method for two *img*s. Dis^{time} denotes the timespan of two timestamps *tp*s. Dis^{dir} denotes the angle of two shooting directions *sDir*s.

2) Generating Data for Experiments

In order to evaluate the performance of PTree, we use synthesized picture streams with varying lengths and temporal densities. These synthesized datasets are generated based on a real application's dataset, as presented below.

FlierMeet [7] is a mobile social network application and 38 students were recruited to use the application. It collected over 2,000 context-tagged pictures from all the students within 8 weeks. A student checking in via photographing at a specific place implies that he is capable of accomplishing the sensing tasks (i.e. taking pictures of interesting fliers) at that place. Five places having a large number of check-ins are chosen to generate a dataset, and the CDF (Cumulative Distribution Function) of these check-ins is shown in Figure 9.

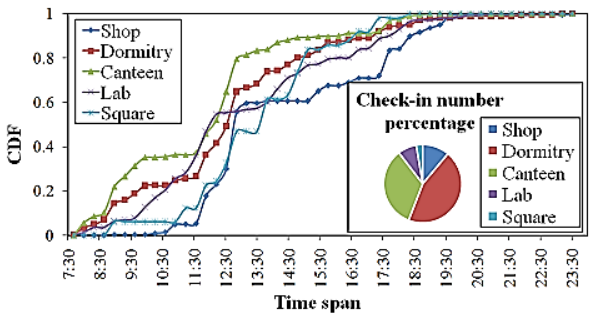


Fig. 9. The CDF of check-in number at different places in FlierMeet dataset.

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

Each simulated picture stream $\mathbb{X}$ has the image similarity matrix $\mathbb{I}$ of pictures in $\mathbb{X}$. Distributions of both timestamps and locations of $\mathbb{X}$ meet distributions shown in Figure 9. The 3D shooting direction of $X_i \in \mathbb{X}$ is randomly created within the range of $[0,2\pi)$ for each dimension. Four features of the data: location (L), image (V), shooting direction (D), and timestamp (T), are adopted in the evaluation, and a 4-tuple <*tp*, *sDir*, *loc*, *img*> denotes the simulated picture. The simulation parameters used for generating $\mathbb{X}$ include *Tsim*, *Dsim*, *Lsim*, *Vsim*, $|\mathbb{X}|$, *TS* and *TE*. These parameters are used according to the following rules: (i) $TS \le tp \le TE$; (ii) Given two pictures X_i and X_j, if $Dis^{time}(tp_i\text{-}tp_j) \le Tsim$, $Dis^{dir}(sDir_i, sDir_j) \le Dsim$, and $Dis^{geo}(loc_i, loc_j) \le Lsim$, then $Dis^{SIFT}(img_i, img_j) \le Vsim$ and $\mathbb{I}_{i,j}$=True (i.e. X_i and X_j are similar). We use the permutations of these four features to generate LMs, such as "T,L,V,D", "T,V,L,D".

C. Simulation-Based Evaluation

1) Performance Evaluation based on PTree Shapes

For the same picture stream, if PTrees are generated with different LMs or different BPs, their shapes might be different. We use two sets of data stream simulation parameters shown in Table IV and a BP $\mathbb{B}_e$={<Dis^{geo}, 40(meter)>, <Dis^{time}, 10(minute)>, <Dis^{dir}, π/4>} for experiments.

TABLE IV
DATA STREAM SIMULATION PARAMETERS.

	TS	*TE*	*Tsim*	*Dsim*	*LSim*
$SimuP_1$	7am	11pm	20(minute)	π/6	20(meter)
$SimuP_2$	10am	11am	20(minute)	π/6	20(meter)

Sixteen PTrees are generated from the same dataset generated based on $SimuP_1$ and $\mathbb{B}_e$. Eight of them are representatively selected and shown in Figure 10. Performance evaluation of selection results based on these PTrees is shown in Table V and Table VI. Experimental results show that *ComC* is related to the PTree shape on one hand. On the other hand, the element-similarity (Eq. (5)) matrix of picture selection results is shown in Table VI. In addition, utility-similarity (Eq. (6)) of each pair of results is almost all at 100%, so we do not show the matrix. As a conclusion, the selection subset will be different if LMs are different, but their utilities are extremely close. So we do not care which picture should be selected but whether these selected pictures have high coverage.

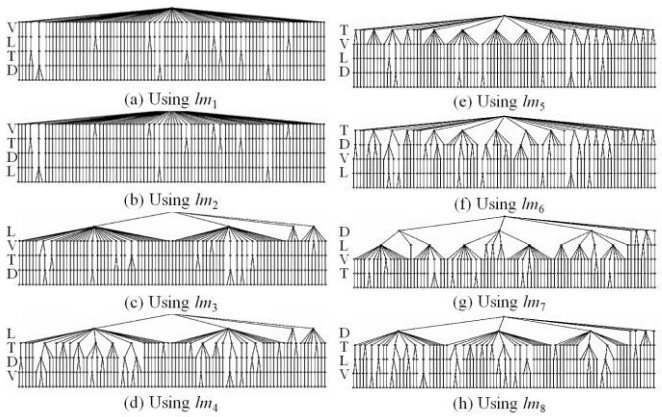


Fig. 10. Shapes of the PTrees generated with the same dataset and different LMs. Here $SimuP_1$ and $\mathbb{B}_e$ are used, and $|\mathbb{X}|$=96. Letters "T, D, V, L" are used to represent LMs, and T denotes the time stamp, D denotes the shooting direction, V denotes the image feature, and L denotes the location. PTrees in (d) (g) (h) have more A-shape subtrees than others PTrees, and their computing costs are less than others too.

TABLE V
EVALUATION OF PTREES AND SELECTION RESULTS WHEN USING DIFFERENT LMS.

No.	LM	*SelRt*(%)	*Covr*(%)	*ComC*	*AComC*	*Redn*(%)
lm_1	V,L,T,D	95.6	100	3,556	39.0	9.2
lm_2	V,T,D,L	95.6	100	3,548	38.9	9.2
lm_3	L,V,T,D	94.5	100	1,337	14.6	10.4
lm_4	L,T,D,V	91.2	98.7	689	7.5	8.4
lm_5	T,V,L,D	92.3	97.4	1,344	14.7	10.7
lm_6	T,D,V,L	91.2	97.4	1,309	14.3	9.6
lm_7	D,L,V,T	92.3	100	676	7.4	8.3
lm_8	D,T,L,V	91.2	100	699	7.6	7.2

$SimuP_1$ and $\mathbb{B}_e$ are used here.

TABLE VI
SELECTION'S ELEMENT-SIMILARITY DEGREE.

LMs	lm_1	lm_2	lm_3	lm_4	lm_5	lm_6	lm_7	lm_8
lm_1	-	1.0	.90	.86	.87	.87	.87	.86
lm_2	1.0	-	.90	.86	.87	.87	.87	.86
lm_3	.90	.90	-	.94	.97	.97	.95	.96
lm_4	.86	.86	.94	-	.94	.94	.96	.97
lm_5	.87	.87	.97	.94	-	1.0	.93	.94
lm_6	.87	.87	.97	.94	1.0	-	.93	.94
lm_7	.87	.87	.95	.96	.93	.93	-	.98
lm_8	.86	.86	.96	.97	.94	.94	.98	-

$SimuP_1$ and $\mathbb{B}_e$ are used here.

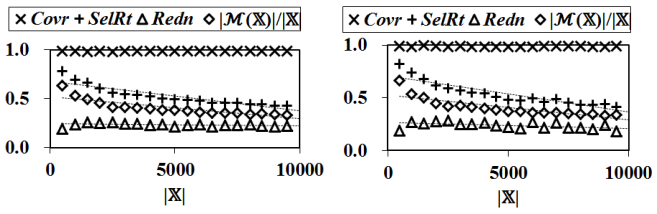
Next, we analyze what shape of the PTree should be chosen. As shown in Table V, computing costs *ComC*s of using different LMs are from 676-3556, and PTrees in Figure 10 shows varying shapes, so we can deduce that nodes in the upper layer should not have a high fan-out degree, which is the characteristic of an A-shape PTree. In addition, using the same BP, iT-shape PTree cannot be generated if either A-shape PTree or I-shape PTree can be generated. Therefore, clustering the picture stream with an A-shape PTree is highly efficient, and A-shape can be achieved by setting proper LMs.

Because $SimuP_1$ uses a large task timespan for creating the simulation dataset, the data distribution is sparse with little redundancy. Therefore, most *SelRt*s are very high in Table V. In the following, we change the sizes and data distribution of the simulated datasets to evaluate PTree's performance.

2) Performance Evaluation of Stability of the PTree

In order to evaluate the flexibility of the PTree-based selection algorithm when $|\mathbb{X}|$ increases, we randomly choose two LMs, lm_4 and lm_6. $SimuP_1$ and $\mathbb{B}_e$ are still used.

The effectiveness of the PTree-based selection is shown in Figure 11 and Figure 12. Experimental results show that the effectiveness of using lm_4="L,T,D,V" or lm_6="T,D,V,L" is similar. When $|\mathbb{X}|$ increases, the values of *SelRt* and $|\mathcal{M}(\mathbb{X})|/|\mathbb{X}|$ decrease, while the values of *Redn* and *Covr* remain nearly steady. *Covr* reaches 100%, and *Redn* stays around 20%.



(a) Using the LM lm_4. (b) Using the LM lm_6.

Fig. 11. The effectiveness of data selection.

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

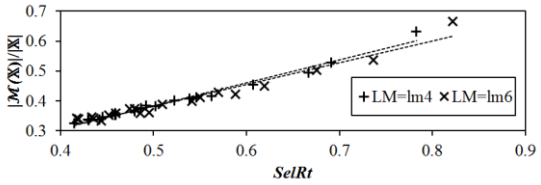
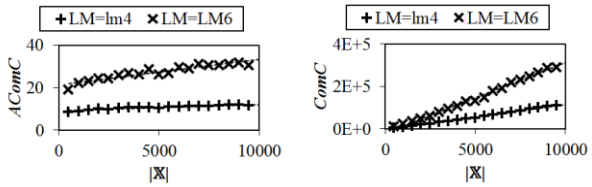


Fig. 12. The positive relationship between *SelRt* and $|\mathcal{M}(\mathbb{X})|/|\mathbb{X}|$.

The efficiency evaluation in Figure 13 proves that computation costs are related to the PTree shape. Because *ComC* is linear with $|\mathbb{X}|$ and *AComC* is the slope, regardless of how long the stream is, each picture will efficiently receive the feedback from the data center. For example, *AComC* is less than 40 in Figure 13 (a), which means that the data aggregator server can give feedback to the worker after less than 40 times simple computations.



(a) Average computing costs. (b) Total computing costs.

Fig. 13. The efficiency evaluation.

3) The Impacts of the Data Density on Picture Selection

Based on the findings discussed above, it is obvious that choosing a proper LM will significantly decrease the computing cost. As shown in Figure 11, *SelRt* decreases when $|\mathbb{X}|$ increases as using the same simulation parameter $SimuP_1$, which implies that more redundant data will emerge when $|\mathbb{X}|$ increases, so it is still a question whether the performance will be impacted by the density or the distribution of data. The dataset used in Figure 10 is very sparse, which can be observed through the sizes of micro clusters, so we simulate a denser dataset with the parameter $SimuP_2$ in Table IV. As shown in Figure 14 and Table VII, although $|\mathbb{X}|$ and LMs are the same with the example in Figure 10, the task timespan (i.e. $|TS\text{-}TE|$) of $SimuP_2$ is shorter than that of $SimuP_1$, so the data density is enlarged and *SelRt* decreases.

TABLE VII
EVALUATION OF PTREES AND SELECTION RESULTS WHEN USING DIFFERENT LMS.

No.	LM	*SelRt*(%)	*Covr*(%)	*ComC*	*AComC*	*Redn*(%)
lm_1	V,L,T,D	71.4	100	2,374	26.0	18.4
lm_2	V,T,D,L	71.4	100	2,362	25.9	18.4
lm_3	L,V,T,D	78.0	100	1,112	12.2	25.3
lm_4	L,T,D,V	76.9	98.1	623	6.8	25.7
lm_5	T,V,L,D	78.0	100	1,181	12.9	25.3
lm_6	T,D,V,L	76.9	100	809	8.8	24.2
lm_7	D,L,V,T	70.3	100	659	7.2	17.1
lm_8	D,T,L,V	72.5	100	612	6.7	19.7

$SimuP_2$ and $\mathbb{B}_1$ are used here.

Because the check-in distribution shown in Figure 9 is different in varied time spans at the same place as well as in the same time span at different places, the spatial and temporal distribution of the entire picture set is uneven. To compare the impacts of the sensing data distributions on the selection performance, we choose three two-hour time spans for *TS* and *TE* of the simulation parameter, i.e., *tSpan*1=7am to 9am, *tSpan*2=3pm to 5pm, and *tSpan*3=5pm to 7pm, for simulation parameters. Based on this varying-density dataset, experimental results in Figure 15 show that $|\mathcal{M}(\mathbb{S})|$ and *SelRt* are tightly related to the temporal and spatial density distribution of pictures, *Redn* is loosely related to it, and *AComC* and *Covr* are barely related to it at all.

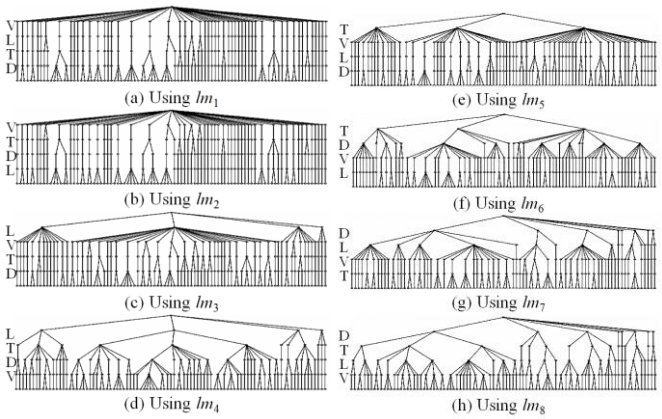


Fig. 14. Shapes of the PTrees generated with the same dataset and different LMs. Here $SimuP_1$ and $\mathbb{B}_e$ are used and $|\mathbb{X}|$=93.

4) Findings

From the simulation-based evaluation, three findings with regard to the efficiency and effectiveness of the PTree-based data selection can be drawn as follows.

a) Although the computation cost of selection increases with the length of the data stream (i.e. $|\mathbb{X}|$), the average computation cost increases rather slowly, even nearly steadily.

b) The size of our selected subset and the size of the MDS of $\mathbb{X}$ have a positive relationship, which means that plenty of pictures are selected to obtain the maximal coverage. The coverage and redundancy of our selected subset are also nearly steady when the data distribution reaches saturation (meaning plenty of data are in all time slots, geography grids, and shooting angle blocks).

c) The efficiency can be assured if branching parameters and layering mapping are properly set to obtain an A-shape PTree.

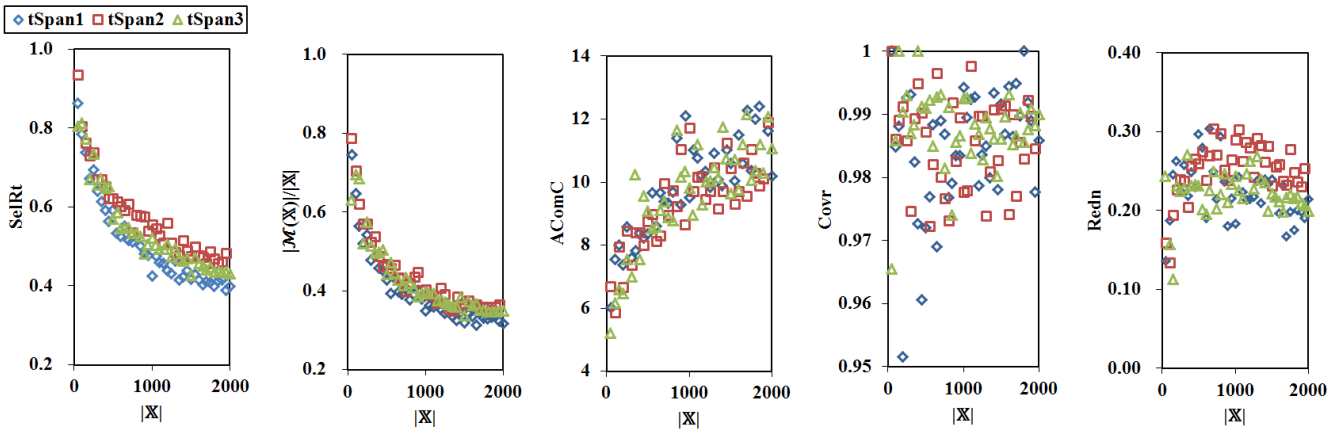
These findings prove that our method has good flexibility to select data when the length of the data stream is generally unknown.

D. Performance Evaluation with the Real Dataset

In order to evaluate the PTree algorithm with the real dataset, we select 1045 pictures of 568 fliers from FlierMeet dataset and attributes of those pictures must include the location, the timestamp, and the shooting direction. The naïve algorithm used by FlierMeet (NFM) can be considered as a one-layer PTree and the unique layer is corresponding to the visual feature *img*. The clustering result of NFM is considered as the ground truth, then *Redn*=0 and *Covr*=1. As shown in Figure 16, we compare the efficiency and effectiveness of NFM and some results based on different PTrees. Experimental results show that all multi-layer PTrees run much quicker than NFM and save 55% to 99% time. The redundancy *Redn* of the multi-layer PTree is higher, but the coverage *Covr* is close to NFM, i.e the

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

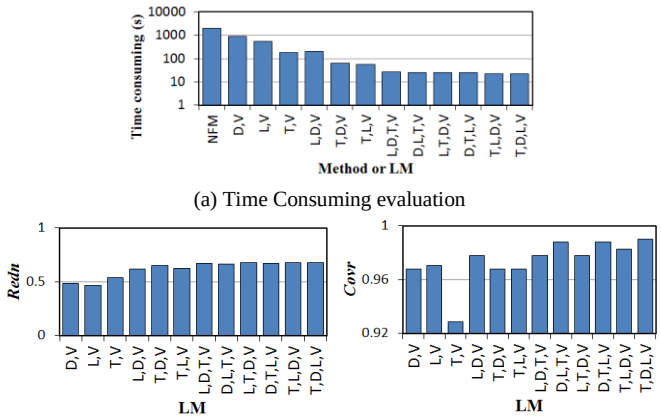
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal



(a) *SelRt* vs. |𝕏| (b) |𝓜(𝕏)|/|𝕏| vs. |𝕏| (c) *AComC* vs. |𝕏| (d) *Covr* vs. |𝕏| (e) *Redn* vs. |𝕏|

Fig. 15. Comparison of the efficiency and effectiveness performance in different time spans.

quality of sensing is assured. Therefore, PTree algorithm increases the efficiency of selection on the premise of increasing redundancy of selected data, how to effectively use PTree depends on the tradeoff between efficiency and redundancy, and how to efficiently use PTree depends on selecting proper features according to the task definition.



(a) Time Consuming evaluation

(b) Redundancy evaluation (c) Coverage evaluation

Fig. 16. Performance comparison of NFM and PTree with varying layers.

E. Discussion

Initial results show that CrowdPic is practical and promising. Though, there are still some limitations to be improved in the future work, as discussed below.

Related modules. There are four key modules of the generic picture collection framework, including task definition, task assignment, picture collection, and incentive mechanism. These four modules are not isolated from each other. For instance, a high-efficient data pre-selection method for different tasks, which is focused in this paper, is utilized to save data requesters' incentive cost because both traffic cost and energy cost are key factors to calculate the incentive rewards to the workers.

Dealing with Ambiguous Task Constraints. Data requesters sometimes cannot predict the distribution or the context of their sensing targets, so they might set ambiguous or wrong task constraints. In order to maintain the quality of sensing, more complex and refined task model is needed, and heuristic task-creating with guidelines will be also helpful for task specification.

Pruning. Since some micro-clusters might not get a new element for a certain period and might become out of date, it is possible to remove these static branches to save the computation cost as a PTree grows. When and how to make branch-cutting are also issue to be focused on in the future.

VII. Conclusion and the Future Work

Selecting highly-relevant data from an evolving picture stream is a fundamental problem for mobile crowd photographing (MCP). In this paper, we have introduced a generic task-driven MCP framework that supports optimal data selection for varied MCP tasks, which are configured based on a multi-facet task model proposed for satisfying all sorts of MCP applications' demands. A pyramid tree-based model is developed to efficiently cluster streaming pictures with its adjustable generation parameters meeting the multiple constraints derived from different MCP tasks. Evaluation results have validated the effectiveness (in terms of sensing coverage and redundancy), efficiency, and flexibility of our method.

Based on the algorithm and the findings in this paper, our future work is as follows. First, we will pay more attention to heuristic task-creating and task-accomplishing with guidelines based on the multi-facet task model to promote the data quality in the picture-taking stage. Second, besides mobile client-server interaction and data selection in the current study, we will study the interaction among local mobile clients and have data selection and fusion at the local side. This will also decrease the cost and efficiency of high-quality data transmission and aggregation.

References

[1] B. Guo, C. Chen, D. Zhang, Z. Yu, Alvin Chin. "Mobile Crowd Sensing and Computing: When Participatory Sensing Meets Participatory Social Media", *IEEE Communications Magazine*, 2016, vol. 54, no.2, pp. 131-137, 2016.

[2] S. Kim, C. Robson, T. Zimmerman, J. Pierce, and E. M. Haber, "Creek watch: pairing usefulness and usability for successful citizen science," in *Proc. of CHI'11*. ACM, 2011, pp. 2125–2134.

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2017.2648860, IEEE Internet of Things Journal

[3] S. Reddy, D. Estrin, M. Hansen, and M. Srivastava, "Examining micro-payments for participatory sensing data collections," in *Proc. of UbiComp'10*. ACM, 2010, pp. 33–36.

[4] Y. Hua, W. He, X. Liu, et al. "SmartEye: Real-time and efficient cloud image sharing for disaster environments," in *Proc. of INFOCOM'15*. IEEE, 2015, pp. 1616–1624.

[5] K. Tuite, N. Snavely, D.-y. Hsiao, N. Tabing, and Z. Popovic, "Photocity: training experts at large-scale image acquisition through a competitive game," in *Proc. of CHI'11*. ACM, 2011, pp. 1383–1392.

[6] J. White, C. Thompson, H. Turner, B. Dougherty, and D. C. Schmidt, "Wreckwatch: automatic traffic accident detection and notification with smartphones," *Mobile Networks and Applications*, vol. 16, no. 3, pp. 285–303, 2011.

[7] B. Guo, H. Chen, Z. Yu, and et al., "FlierMeet: A mobile crowdsensing system for cross-space public information reposting, tagging, and sharing," *Mobile Computing, IEEE Transactions on*, vol. 14, no. 10, pp. 2020–2033, 2015.

[8] Y. Jiang, X. Xu, P. Terlecky, T. Abdelzaher, A. Bar-Noy, and R. Govindan, "Mediascope: selective on-demand media retrieval from mobile devices," in *Proc. of IPSN'13*. ACM, 2013, pp. 289–300.

[9] J. Dong, Y. Xiao, M. Noreikis, et al. "imoon: Using smartphones for image-based indoor navigation," in *Proc. of SenSys'15*. ACM, 2015, pp. 85–97.

[10] S. Morishita, S. Maenaka, D. Nagata, et al. "SakuraSensor: quasi-realtime cherry-lined roads detection through participatory video sensing by cars," in *Proc. of UbiComp'15*. ACM, 2015: 695–705.

[11] J. Goldman, K. Shilton, J. Burke, and etc., "Participatory sensing: A citizen-powered approach to illuminating the patterns that shape our world," *Foresight & Governance Project, White Paper*, pp. 1–15, 2009.

[12] M.-R. Ra, B. Liu, T. F. La Porta, and R. Govindan, "Medusa: A programming framework for crowd-sensing applications," in *Proc. of MobiSys '12*. New York, NY, USA: ACM, 2012, pp. 337–350.

[13] H. Chen, B. Guo and Z. Yu. "CooperSense: A Cooperative and Selective Picture Forwarding Framework based on Tree Fusion", *International Journal of Distributed Sensor Networks*, 2016.

[14] A. Zaslavsky, P. P. Jayaraman, and S. Krishnaswamy, "Sharelikescrowd: Mobile analytics for participatory sensing and crowdsourcing applications," in *Proc. of ICDEW'13*. IEEE, 2013, pp. 128–135.

[15] B. Guo, H. Chen, Q. Han, Z. Yu, D. Zhang and Y. Wang. "Utipay: worker-contributed data utility measurement for visual crowdsensing systems," *IEEE Trans. on Mobile Computing*, 2016.

[16] D. Zhang, L. Wang, H. Xiong, and B. Guo, "4w1h in mobile crowd sensing," *Communications Magazine*, IEEE, vol. 52, no. 8, pp. 42–48, 2014.

[17] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *Communications Magazine*, IEEE, vol. 52, no. 8, pp. 29–35, 2014.

[18] Y. Wu, Y. Wang, W. Hu, et al. "Resource-aware photo crowdsourcing through disruption tolerant networks," in *Proc. of ICDCS'16*. IEEE, 2016, pp. 374–383.

[19] D. Zhang, H. Xiong, L. Wang, and G. Chen, "Crowdrecruiter: selecting participants for piggyback crowdsensing under probabilistic coverage constraint," in *Proc. of UbiComp'14*. ACM, 2014, pp. 703–714.

[20] B. Guo, H. Chen, Z. Yu, W. Nan, X. Xie, D. Zhang and X. Zhou. "TaskMe: toward a dynamic and quality-enhanced incentive mechanism for mobile crowd sensing," *International Journal of Human-Computer Studies*, 2016.

[21] E. Koukoumidis, L.-S. Peh, and M. R. Martonosi, "Signalguru: leveraging mobile phones for collaborative traffic signal schedule advisory," in *Proc. of MobiSys'11*. ACM, 2011, pp. 127–140.

[22] S. Sehgal, S. S. Kanhere, and C. T. Chou, "Mobishop: Using mobile phones for sharing consumer pricing information," in *Demo Session of the Intl. Conf. on Distributed Computing in Sensor Systems*, 2008.

[23] Y. Wang, W. Hu, Y. Wu, and G. Cao, "Smartphoto: a resourceaware crowdsourcing approach for image sensing with smartphones," in *Proc. of MobiHoc'14*. ACM, 2014, pp. 113–122.

[24] H. Chen, B. Guo, Z. Yu, S. Huangfu, W. Nan, and W. Wu, "Crowdpic: An interactive and selective picture collection framework for participatory sensing systems," in *Proc. of CIT'14*. IEEE, 2014, pp. 512–519.

[25] D. Mizell, "Using gravity to estimate accelerometer orientation," in 2012 *16th International Symposium on Wearable Computers*. IEEE, 2003, pp. 252–252.

[26] R. Balakrishnan and K. Ranganathan, *A textbook of graph theory*. Springer, 2012.

[27] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[28] H. Chen, B. Guo, Z. Yu and Q. Han, "Toward real-time and cooperative mobile visual sensing and sharing," in *Proc. of INFOCOM'16*. IEEE, 2016, pp. 1-9.

Huihui Chen is a doctoral candidate from Northwestern Polytechnical University, China. She received her M.Eng in computer science from Zhengzhou University, China in 2006 and B.Eng in computer science from Northeast Dianli University, China in 2000. Her research interests include ubiquitous computing and mobile crowd sensing.

Bin Guo is a professor from Northwestern Polytechnical University, China. He received his Ph.D. degree in computer science from Keio University, Japan in 2009 and then was a post-doc researcher at Institut TELECOM SudParis in France. His research interests include ubiquitous computing, mobile crowd sensing, and HCI.

Zhiwen Yu is a professor and vice-dean at the School of Computer Science, Northwestern Polytechnical University, China. He has worked as an Alexander Von Humboldt Fellow at Mannheim University, Germany from Nov. 2009 to Oct. 2010, a research fellow at Kyoto University, Japan from Feb. 2007 to Jan. 2009. His research interests cover pervasive computing and HCI.

Liming Chen is a professor of Computer Science and the Head of the Context, Intelligence and Interaction Research Group (CIIRG) in the School of Computer Science and Informatics at De Montfort University, United Kingdom. He received his B.Eng and M.Eng from Beijing Institute of Technology, Beijing, China, and his Ph.D. in Artificial Intelligence from De Montfort University, UK. His research interests include activity recognition, intelligent systems, smart environment and assisted living.

Xiaojuan Ma is an assistant professor of Human-Computer Interaction (HCI) at the Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST). She received the Ph.D. degree in Computer Science at Princeton University. Her research interests include human-computer interaction, crowdsourcing, ubiquitous computing, and computational linguistics.

2327-4662 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.